Applying the Concept of Figural Goodness for Automatic Design of Dataset Visualization

T. RZEŹNICZAK
tomek.rzezniczak@gmail.com

Institute of Computer and Information Systems
Faculty of Cybernetics, Military University of Technology
Kaliskiego Str. 2, 00-908 Warsaw, Poland

Many approaches to automatic design of graphical presentation are focused on creating graphical presentation for a given data type, often without considering actual instances of data. This is due to the fact that they aim to be general purpose solutions. In this study a method which draws attention to characteristics of the actual data to be presented is adopted. Knowing characteristics of data that will be shown to the observer in advance, allows much more customized approach and better performance of constructed visualization. Visualisation performance in this case is understood as an ability to identify and recognize presented objects quickly and easily. Construction of the method relies heavily on research in the area of data visualization and perceptual psychology – with special emphasis on *figural goodness*.

Keywords: data visualization, graphical languages, structural information theory, figural goodness.

1. Introduction

Data visualization is nowadays a common way to represent and analyze information. There are many well-known benefits of data visualization, among others: human natural ability to quickly process visual patterns, large amount of information represented in a single image, understanding of complex data is assisted. All of these results in great interest in the subject, so there is nothing surprising in many works being carried out on automatic design of graphical presentations [20, 23]. Their most common goal is to build a framework which creates a presentation method for a given data type. In their assumptions the data type is unrestricted. That means these are general purpose solutions that can present any kind of information. This gives great flexibility, but on the other hand surely there must be better solutions in some cases.

In this paper the space of data types for which a presentation is constructed is narrowed, this results in the ability to use more informed design. The data considered here is a set of objects, each with limited set of attributes (more precise definition will be provided later). Other narrowing assumption is that a presentation is constructed for well-defined purpose. The goal in this case is to identify and recognize presented object among others in the set. This is a very common human task, especially for diagnostic processes, where a number of characteristics describing an actual state of some entity exist. These characteristics enable to recognize a cause of the entity's condition. A real-world example is medical diagnosis: referring to the process of identifying a possible disease based on symptoms [1, 2]. Here, the visualized objects correspond to the condition being diagnosed and the presentation goal is to support this process.

In the next section an attempt to outline the problem of constructing visual presentation to achieve the above goal will be undertaken. But in the beginning, in order to set some theoretical background, areas of: *graphical languages*, *Gestalt psychology* and *Structural Information Theory*, will be introduced [9, 11, 20, 21, 23, 27] .

2. Graphical languages

A graphical language can be considered as a description of how a graphic corresponds to represented information. According to pioneers in the area of automatic design of graphical presentations, graphical languages are similar to other formal languages through defining precise semantic and syntactic of constructed sentences [20]. Therefore, presentations can be seen as sentences in a graphical language. In context of an input data set, the main characteristics of a graphical language are its: expressiveness and effectiveness.

The expressiveness criteria can be clearly specified for each graphical language. A graphical language satisfies expressiveness criteria when it is possible to generate sentences that express all the facts represented by the input data. The issue here is that the generated sentences may introduce additional facts which are not correct. Therefore, a language is expressive when it represents all the input facts and these facts only.

The effectiveness criteria is more complicated to evaluate, since it can base on many different factors, like accuracy of interpretation, its speed, presentation simplicity or any other characteristic related to human perception. For example, Mackinaly in his work on automating presentation design focuses on accuracy of interpretation [20]. He based his study on the fact that a person has to perform perceptual task when interpreting a graphical presentation. The task strongly depends on a graphical technique used to construct the presentation. Determining a position of graphical object, its length, color, size, etc., may be an example of such task. Since human perception is equipped for some tasks better than the other, Mackinaly measured effectiveness comparing perceptual task required by different graphical languages. For this purpose, Mackinaly prepared a ranking (Table 1) of perceptual tasks' efficiency depending on the type of graphical coding. Mackinlay's ranking includes *quantitative*, *nominal* and *ordinal* data types [7, 20, 25].

Tab. 1. Mackinlay's ranking of perceptual tasks efficiency − from the most to the least efficient [20]

Quantitative	**Ordinal**	**Nominal**
Position	Position	Position
Length	Intensity/Value	Colour (hue)
Angle	Colour (saturation)	Texture
Slope	Colour (hue)	Connection
Area (Size)	Texture	Containment
Volume	Connection	Intensity/Value
Intensity/Value	Containment	Colour (saturation)
Colour (saturation)	Length	Shape
Colour (hue)	Angle	Length
Texture	Slope	Angle
Connection	Area (Size)	Slope
Containment	Volume	Area (Size)
Shape	Shape	Volume

Looking for optimal language to represent given set of data, it is necessary to identify a space of possible graphical languages. Mackinaly limited the space to a set of well-defined primitive languages like: line chart, bar chart, tree, vertical/horizontal axis, color, size, shape, etc. The primitive languages are classified by their encoding techniques, for instance: *single position language* (vertical/horizontal axis) encodes information by its position, *retinal list languages* (color, size, shape, etc.) encode information using six retinal properties defined by Bertin [4, 5]. Since the usage of a single primitive language may be not enough for many possible presentations' needs, Mackinaly introduced composition algebra. The algebra consists of primitive languages and composition operators which allow producing alternative languages based on primitive ones.

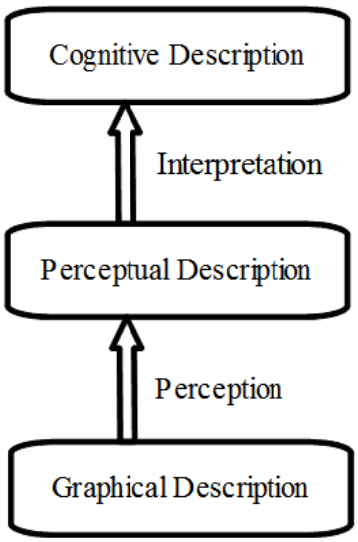

Fig. 1. Model of graphical presentation understanding [23]

Despite the fact that, the composition algebra extends capabilities of Mackinaly's solution, it is still limited because of small set of primitive techniques. *First-principle framework* constructed by Michael Shiff, represents another approach to the topic of automating presentation design [23]. Deep decomposition of graphical languages design space and leveraging human visual perception process for selection of optimal presentation method is the main characteristic of *first-principle*. Schiff used perception model similar to those presented by Pinker's theory of graph understanding and Lohse's model of extracting information from graphs [17, 18, 22]. *First-principle* bases on the assumption that two steps: perception and interpretation, take place independently. Firstly, human perception decodes given presentation and subsequently interpretation of perceived

information occurs. The presentation can be seen as a graphical description composed of graphical objects which are identified by perception and based on it a perceptual description is built. The perceptual description is then interpreted and transformed to a cognitive level. Cognitive description is a mental representation of information incorporated in a graphical presentation. The basic model of perceptual processing and information extraction is shown in Figure 1.

Coming back to the size of design space, in Schiff's approach, graphical description is more fine-grained than Mackinaly's set of primitive languages. Graphical objects that make up the graphical description are instances of graphical primitives. Possible primitives can be: lines, arrows, circles, rectangles. Each primitive has its fixed set of parameters, like: color, size, position, which altogether give large design space. To cope with this and handle construction of graphical language, Schiff defines two types of principles: logical principles and psychological principles [23]. Logical principles are a kind of filters which determine if a language is able to represent input data. Psychological principles perform further selection based on how suitable the language is, considering people accuracy and speed in correct understanding of encoding convention (encoding convention is elaborated in next sections). This is necessary since *first-principle* may have to deal with previously unknown graphical languages, contrary to Mackinaly's approach [20].

Psychological principles are subsequently divided into interpretive principles and perceptual principles. Interpretive principles are most difficult to specify objectively, because they describe characteristics that make a presentation easy to understand without any confusion. On the other hand, the perceptual principles determine how effective a graphical language is for a given data, where effectiveness is measured by speed and accuracy of perceptual distinction made by observer of a graphical presentation. Comparing to interpretive principles, this area is far more explored by researches and has better theoretical foundations.

Summarizing, a set of principles is used for the purpose of filtering and selecting of an optimal graphical language for input data type. The finally selected language in terms of *first-principle* approach is composed of:

- a set of graphical primitives that will be used in the presentation
- a specification of correspondence between domains of each data tuple and some properties of the graphical primitives
- a specification of relationship between graphical primitives required to encode data relation, when multiple primitives encode single data relation.

3. Selecting encoding convention

Now, let's focus on how to build a graphical language and eventually transform information into its graphical representation. As mentioned earlier, a graphical presentation can be described at several levels. The method of transition from one level to another is called *encoding convention* [23] – starting from the cognitive description, through the perceptual description down to the graphical description. Some examples of entities used at different levels are presented in Tab. 2. It is evident that *encoding convention* must specify a systematic method of mapping between subsequent description levels for a certain type of data set. Selected method should allow constructing a graphical presentation for any data set instance of given type. Automating the selection process is a critical area for the domain of automatic presentation design. In this article the area will be narrowed, so the *encoding convention* is partially predefined in a manual process, at the same time the focus is aimed at particular data type and detailed characteristics of the selected encoding convention.

Tab. 2. Examples of entities for different levels of description [23]

Level	**Entitites/Components and examples**
Cognitive	objects/tuples/relations (e.g. <attrib1, attrib2, ...>) domains (set of possible attributes)
Perceptual	Perceptual objects (e.g. circle, rectangle) Properties (e.g. size, postion)
Graphical	Graphic objects (e.g. circle, point) Parameters (e.g. x, y, radius)

The task of creating graphical representation is naturally divided into two subtasks which rely on the introduced description levels. The first one is to transform from the given data set (which is actually a cognitive description) to perceptual model. The main issue here is to choose data representation which allows an observer to note

all semantically relevant attributes in such a way, that information can be extracted. The second subtask is to consider how the chosen perceptual description should be instantiated into graphical objects with their layout and properties, so as to ensure that the recipient is able to extract certain perceptual properties and perceptual relations effectively. Described subtasks were called by Shiff respectively: *representation* and *layout.*

Before explaining a proposed *encoding convention,* it is necessary to introduce data type which will be explored and the goal of the graphical presentation. The data type is defined as a set of objects; each object has a set of attributes where all attributes belong to a single domain. The domain type is *nominal* and the number of attributes for particular object is not explicitly defined but depends on the instance of the data set. Following this definition, the instance of information to be presented can be modelled as a set of objects O and a set of possible domain attributes A, where each object $x \in O$ is defined as $x \subseteq A$. The main purpose of the presentation is to enable the observer to identify individual object x and extract information about its characteristics. In other words, *encoding convention* must allow a graphical representation and differentiation of each object from the set O.

It must be noted, that selected e*ncoding convention* is being prepared for one specific data type (and even for a particular instance of data) – not for any data type, therefore the *representation* stage can be constructed once and it can be done manually, taking into account human perceptual capabilities. The *layout* task is treated differently and engages automated processes for solving the problem, what will be elaborated further in the article.

On the basis of earlier definitions, the proposed encoding convention can be described as follows:

- cognitive level description – a set of objects O, a set of possible domain attributes A, each object $x \in O$ is defined as $x \subseteq A$
- perceptual level description – perceptual objects are: *points* and *lines*
- mapping between cognitive and perceptual level – each attribute of the cognitive object x is represented by a single point on a plane, all points that belong to one object are connected by lines, so the whole object creates a closed polygon
- graphical level description – a point on grid of n equal cells, where n is a number of different attributes in the set A. Each cell represents a different attribute and a dot can be placed only inside a cell
- mapping between perceptual and graphical level – each point of the perceptual level is represented by a single dot on the grid, each line is represented by single line drawn between dots.

Example presentation using this encoding convention can be found in Figure 2. Having above definition, the key problem to solve here is *layout* – how to place points and lines on the grid in such a manner that cognitive objects are effectively extracted, identified and remembered by an observer.

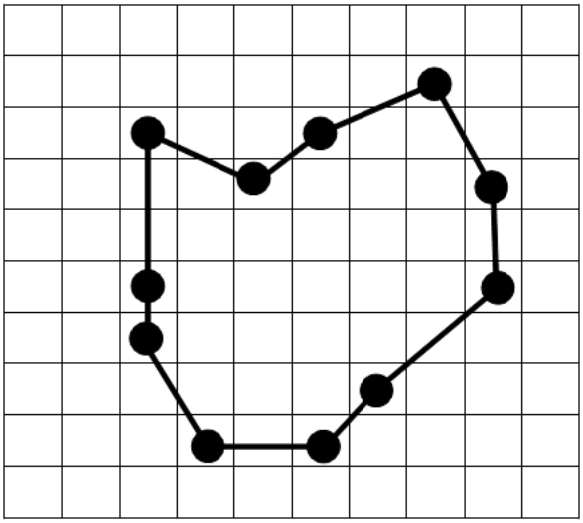

Fig. 2. Example presentation – generated with proposed encoding convention

The *layout task* will be elaborated later in this article. Now I will focus on the rational behind this encoding convention. There is at least a couple of reasons which originate from the data type characteristics.

First of all, objects are described by *nominal* attributes, which means that they are only named values. There is no defined order and no arithmetic operations are possible [25]. The only thing that can be done over *nominal* values is comparing them with other *nominal* values. This implies that standard techniques of visualization such as line charts, bar charts, etc. are hard to apply.

Secondly, looking at Mackinlay's ranking of perceptual tasks efficiency (Table 1), it can be noticed that determining a position of graphical object is most efficient for all the data types including *nominal.* It is clear, that the use of space is the most significant aspect of visualization. Space is treated in a particular way in relation to other image attributes – it is the basis, on which other elements are distributed

[6]. The empty space of an image is a container with a metric structure that can be described by axis. For the *nominal* data type axis are covered by regions divided into subregions. Considering the above arguments, leveraging the use of arbitrary location in space to exhibit attributes of an object, as it is proposed in this study, seems to be well justified.

There is one more argument that cannot be neglected. Such encoding convention allows introduction of shape perception. From all other properties of a graphical object, shape is probably the most important. This is due to the fact that it is the most informative property, which allows to receive a greater number of facts about the object than anything else [21]. As the purpose of presentation specified earlier, is to identify depicted object, taking advantage of object's figure is strongly legitimate. Influence of shape on visual perception of figures will be elaborated further in the next section.

4. From figural goodness to Structural Information Theory

A special sensitivity of human vision to ’good’ structures was first noticed by the authors of *Gestalt psychology*. They claimed that people tend to organize visual elements into groups [27]. Only global relation, between elements forming an image, determines the key aspect of perception. That is why when looking at a set of dots outlining a rectangle, we see a figure not just a set of dots. Max Wertheimer and his co-workers formulated principles explaining perceptual organization called *gestalt laws* [27]. Another achievement of their work is the concept of *figural goodness* [14, 21]. It is based on the observation that some sets of elements seem to be simpler and better organized, so called *good figures*, whereas other sets of elements are complex and poorly organized – *bad figures*. The *figural goodness* characteristic of an object can be well described as a composition of regularity and simplicity. For some shapes such as a circle, any change can only degenerate its *goodness* [21].

The interest in *good figures* is motivated by the efficiency of processing. There is a strong correlation between performance of some tasks and human opinion on the *goodness* of the figure [8]. Garner found evidence that people remember, describe, match and learn *good figures* much better than *bad* ones.

Now let’s focus on the method of how to determine the *goodness* of a figure. Unfortunately, gestalt psychologists did not define any objective measure. A simple method would be to analyze a structure of a figure, for example the number of sides increases perceived complexity. But even for the same number of sides, figure’s complexity can differ (Figure 3) [21]. It appears that the number of components is not an adequate measure and the complexity must depend on the way the components are assembled together. Which comes back to the main assumption of *Gestalt psychology* – global relation of the perceived components is the most important.

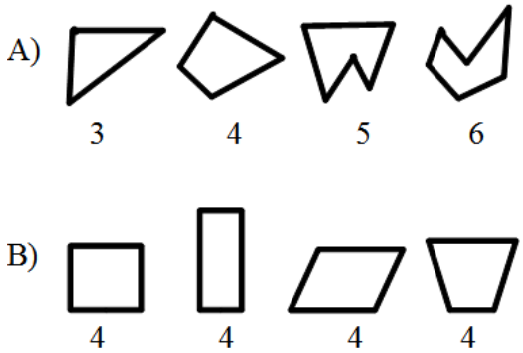

Fig. 3. A) Example on how the number of sides increases perceived complexity;
B) Example on how regularity affects *figural goodness*, even for the same number of sides [21]

There are at least two approaches that take into account the whole structure of a figure in determining complexity. First was formulated by Wendell Garner [8]. His *Rotation and Reflection Subset* theory defines eight transformations applicable to a figure: four rotations (0, 90, 180, 270 – degrees) and four reflections (horizontal, vertical, left-diagonal, right-diagonal – axes). In this approach *goodness* is inversely proportional to the number of different figures generated by the application of all transformations to the original figure. For instance, applying the transformations on a square gives in all eight versions the same square. At the same time, applying it to random trapezoid produces eight different variants of the figure. This simple experiment is consistent with the gestalt assumptions, because a square is an objectively simpler figure then trapezoid. The only problem with transformational approach is that not all cases can be explained. For various figures the number of different transformations may be equal, but they are not equally ’good’ [21].

Other theories which deal well with the previous issue also exist. They are derived from another major gestalt principle formulated

by Koffka – *Law of Pragnanz* [15]. The law indicates that human perception is led by simplicity. Therefore, a preferred interpretation of a shape is reflected in the simplest description of that shape. The example in Figure 4 helps to understand this dependency. *The Pragnanz law* does not specify how to determine the simplicity of the shape. First steps towards finding the method were made thanks to Shannon's *Information Theory* [24].

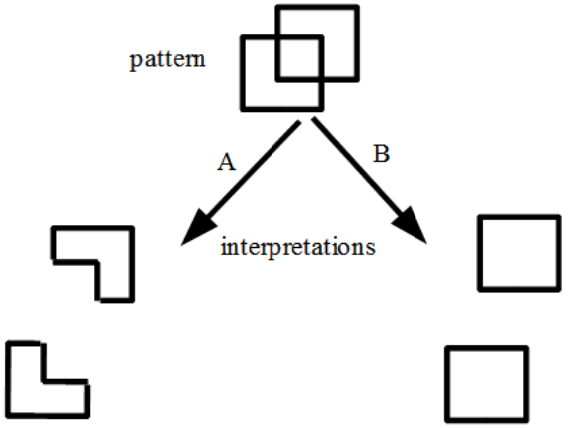

Fig. 4. *Pragnanz law* visualization, one pattern can have many interpretations, but the simplest is preferred. In this case interpretation B is more probable than A [13]

Attneave and McAlister noticed that human visual systems encode shapes optimally by leveraging regularities such as: symmetry or repetition [3, 14]. Many perceptions are possible, but the one actually perceived has the shortest encoding, because each regularity means less bits of information needed for the code. This was a first objective measure of simplicity. But how the code should look like was not specified, until Leeuwenberg introduced *Structural Information Theory* (SIT) [16]. Formalization of SIT encoding model started from 1D and then was generalized to 2D patterns. According to SIT, the process is composed of three steps [12, 13]:

1. 2D visual pattern is represented by a 1D sequence of symbols (e.g. characters), the only restriction is that the 2D pattern is unambiguously reproducible from the sequence of symbols.
2. The sequence of symbols is encoded by means of encoding rules, which describe regularities among symbols and subsequences of symbols.
3. The selection of simplest code is made using complexity metric called *Information Load*, which must be in line with the discussed earlier *Law of Praegnanz*.

Tab. 3. Coding rules examples [13]

Rules	**Symbol sequ.**	**Code**
Iteration	aaa...aa	m*(a)
Symmetry	abcdcba	S[(a)(b)(c),(d)]
Alternation	abacad bacada	<(a)>/<(b)(c)(d)> <(b)(c)(d)>/<(a)>

The encoding rules are strongly connected with visual regularities. SIT developed a set of regularities, which then was narrowed to three major so called transparent holographic regularities. The holographic regularities basically cover all the other regularities (broader rational is beyond the scope of this article), namely: iteration, symmetry, alternation. Rules representing them are respectively: I-form, S-form, A-form, referred together as ISA-forms (Table 3).

Applying the rules to the original symbol sequence and all the subsequences gives combinatorial explosion of the possible minimal end codes. To search this space and evaluate possible coding, one needs mentioned earlier, *information load* metric. Since SIT complexity metric is a separate issue, for the purpose of this article number of symbols that remain in the encoded sequence can be a good enough approximation. For example, below is an approximated *information load* of following codes:

- ababab ↔ 3*((a)(b)) : inf. load = 2
- abcdcba ↔ S[(a)(b)(c),(d)] : inf. load = 4
- abacad ↔ <(a)>/<(b)(c)(d)> : inf. load =3

Summarizing, SIT passes from theory to practice and gives the ability to calculate *figural goodness*. Hence, comparing efficiency of different figures is also feasible.

5. SIT and Layout Task

SIT does not provide a specific way how to transform 2D pattern/figure into 1D sequence of symbols. The natural assumption is that it must be a reversible process. The most common conversion method is based on following a figure contour and coding it as a sequence of sides length and angles between them – *contour code*. Example for a trapezoid is presented on Figure 5, representing it as a code, using contour coding, could take a form of *aebeadcd* (*inf. load* = 8). After applying ISA-form, its shorten version, taking into account symmetries of the trapezoid is: *S*[(*a*)(*e*),(*b*)]*S*[(*d*),(*c*)](*inf. load* = 5).

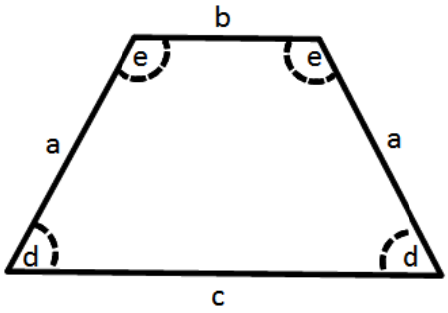

Fig. 5. The figure of trapezoid with labelled sides and angels. 1D sequence could be: *aebeadcd*, and applying ISA-forms: *S*[(*a*)(*e*),(*b*)]*S*[(*d*),(*c*)]

Using the discussion on *graphical languages* and *figural goodness*, let's move to describe the *layout* task. Recalling the task in the *figural goodness* nomenclature, it concerns finding a method of how to arrange the grid of attributes *A* in such a way that all visualized objects *O* form *good figures*. So that it allows effective extracting, identifying and remembering them.

Now, let's consider how to find optimal grid arrangement. Using methods presented in previous sections, it is possible to calculate the information load of a given object for a single grid arrangement. An aggregated complexity measure for the given grid can be obtained by calculating and adding *information loads* of all objects from *O*. The *aggregated information load* is then a selection criteria of the grid arrangements. The lower load the better grid (the grid optimality is inversely proportional to the load). Grid selected this way 'holds a promise' of each object visualisation being a *good figure*.

Although this article has rather an introduction nature and detailed algorithm is not a subject here, I will try to show the list of problems which must be solved in order to complete the *layout task*. Looking for the optimal grid, computational complexity become very crucial point, there are three major problems:

Problem I – figure constructing

As it was mentioned earlier during the discussion on encoding convention, the visualized objects are represented by points and connecting them lines in such a way that altogether they form a closed polygon. In theory, any point can be connected to any line. As it can be observed, the number of possible connections configuration depends on the number of object attributes and it is of order *k*!. Not all of them are valid configurations since intersecting lines are not allowed. Selection criteria among valid polygons are based on spatially contiguous condition, proposed by Tuijl and Leeuwenberg [13, 26]. The computational complexity of a naive algorithm solving this task is $O(k!)$.

Problem II – minimal code

When a figure is constructed it is easy to produce the *contour code*. The next step is to find the minimal code for this *contour code*. Basic approach to this problem may look as follows:

1. Searching for ISA-form which can describe some symbol subsequences;
2. Replacing the founded subsequence with related ISA-forms;
3. Continuing searching in the resulting sequence.

The search space for minimal code grows exponentially with the number of symbols in the initial code. The exponential scale of this task has two sources [10, 12, 13]. In a code of length *N*, the number of different length subsequences is $N*(N+1)/2$, which gives polynomial search space, but all possible combinations of subsequences which make $O(2^N)$ are in our actual interest. The second problem here is finding ISA-forms covering for each subsequence, which is a super-exponential problem $O(2^{NlogN})$.

Problem III – grid selection

Problem I and Problem II are only considering single figure and single grid, but the main task is to check all possible grids. It is easy to calculate that for *n* attributes in the set *A*, *n*! valid grid arrangements exist (assuming a fixed number of rows and columns in the grid – a square grid). Manipulating the ratio of rows to columns gives completely new arrangements and obviously increases the number of all possible combinations. For simplicity reasons only square grids will be considered, even though the problem has complexity of $O(n!)$.

Considering potential solutions to the above problems, *Problem I* can be represented using graph theory. Assuming that points are vertexes in a complete graph, a search for a valid polygon may be approximated by finding minimal Hamiltonian cycle in such a graph [13]. This is a well-known problem for which many heuristic algorithms that can be applied exist.

Regarding *Problem II*, P. van der Helm proposed transparallel processing approach for minimal SIT code search that uses hyperstring concept and achieves acceptable complexity of $O(N^{3+logN})$ [10].

The biggest issue in solving *layout task* comes from *Problem III* – grid selection. It is evident from the above discussion, that using naive approaches and checking all potential grids for a large set of attributes *A* is not possible in finite time (e.g. 100 attributes gives around 10^{157} different grids). This problem does has not had yet good solution and will be a major topic of further research.

6. Conclusions

This article aims to outline the problem of data visualization, where a method of visualization is built for a specific data, not only for a data type but also for an instance of data. This allows to use their characteristics and build otherwise inaccessible solutions. At this stage the use of the data instance is not fully leveraged. It is implicitly covered in the whole presented method, which expresses in the fact that the data set is known from the beginning, but still it is not used directly in a specific algorithm. For example any particular characteristic of an object *o* from *O* is never considered. This leaves room which can be explored in grid selection algorithm. Since the naive solution to this problem is unacceptable, the work must be focused on other methods. Heuristic methods seem to be a good candidate to make use of the data characteristics. For example dividing the grid selection problem into smaller subproblems is one of a potentially possible approach. The division may be set based on occurrence frequency of particular attributes in visualized objects, taking into account similar groups of attributes shared by different objects or any other pattern appearing in data. Knowing such patterns allows to extract them and for instance to prepare fixed solution (fixed *good figure*) for the subset of attributes and objects. Then, it can be used as a constant element of the grid, which reduces the number of possible grid arrangements. This and other heuristics will be the subject of further research.

7. Bibliography

[1] A. Ameljańczyk, *Multiple Optimization*, Warszawa, 1986.

[2] A. Ameljańczyk, ”Properties of the Algorithm for Determining an Initial Medical Diagnosis Based on a Two-Criteria Similarity Model”, *Biuletyn Instytutu Systemów Informatycznych*, Nr 8, 9–16, (2011).

[3] F. Attneave, ”Some informational aspects of visual perception”, *Psychological Review*, 61, 183–193 (1954).

[4] J. Bertin, *Graphics and Graphic Information-Processing*, Walter de Gruyter, Berlin, 1981.

[5] J. Bertin, *Semiology of Graphics: Diagrams*, *Networks*, *Maps*, The University of Wisconsin Press, Wisconsin, 1983.

[6] S. Card, J. Mackinlay, B. Shneiderman, *Readings in information visualization: using vision to think*, Morgan Kaufmann Publishers, San Francisco, 1999.

[7] W. Cleveland, R. McGill, ”Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods”, *Journal of the American Statistical Association*, 79, (1984).

[8] W. Garner, *The processing of information and structure*, MD: Lawrence Erlbaum Associates, 1974.

[9] E. Goldmeier, ”Similarity in Visually Perceived Forms”, *International Universities Press*, Inc., 1972.

[10] P. van der Helm, ”Transparallel processing by hyperstrings”, *Proceedings of the National Academy of Sciences USA*, 101 (30), 10862–10867, 2004.

[11] P. van der Helm, E. Leeuwenberg, ”Accessibility: A Criterion for Regularity and Hierarchy in Visual Pattern Codes”, *Journal of Mathematical Psychology*, 35, 151–213 (1991).

[12] P. van der Helm, E. Leeuwenberg, ”Avoiding Explosive Search in Automatic Selection of Simplest Pattern Codes”, *Pattern Recognition*, Vol. 19, No. 2, 181–191 (1986).

[13] P. van der Helm, E. Leeuwenberg, ”Goodness of Visual Regularities: A Nontransformational Approach”, *Psychological Review*, Vol. 103, No. 3, 429–456 (1996).

[14] J. Hochberg, E. McAlister, ”A quantitative approach to figural goodness”, *Journal of Experimental Psychology*, 46, 361–364 (1953).

[15] K. Koffka, *Principles of Gestalt psychology*, New York: Harcourt, Brace, 1935.

[16] E. Leeuwenberg, ”A perceptual coding language for visual and auditory patterns”, *American Journal of Psychology*, 84(3), 307–349 (1971).

[17] G. Lohse, *A cognitive model for the perception and understanding of graphs*,

The University of Michigan dissertation, 1991.
[18] G. Lohse, ”A cognitive model for the perception and understanding of graphs”, *Human-Computer Interaction*, 8(4), 353–388 (1993).
[19] A. MacEachren, *How Maps Work*, *The Guilford Press*, New York, 1995.
[20] J. Mackinlay, ”Automating the Design of Graphical Presentations of Relational Information”, *ACM Transactions on Graphics* , Vol. 5, Issue 2 (1986).
[21] S. Palmer, *Vision Science: Photons to Phenomenology*, MIT Press, 1999.
[22] S. Pinker, ”A theory of graph comprehension”, *Artificial intelligence and the future of testing*, Lawrence Erlbaum Associates, 1990.
[23] M. Schiff, *Designing graphic presentations from first principles*, Doctoral Dissertation, University of California, Berkeley, 1998.
[24] C. Shannon, W. Weaver, *The mathematical theory of communication*, Urbana: University of Illinois Press, 1949.
[25] S. Stevens, ”On the Theory of Scales of Measurement”. *Science*, 103 (2684), 677–680 (1946).
[26] H. van Tuijl, E. Leeuwenberg, ”Perceptual interpretation of complex line patterns”. *Journal of Experimental Psychology: Human Perception and Performance*, 6, 197–221 (1980).
[27] M. Wertheimer, ”Gestalt theory”, *A sourcebook of Gestalt psychology*, 1–11, New York: The Humanities Press, 1924/1950.

Zastosowanie koncepcji „figural goodness” do automatycznego projektowania wizualizacji zbioru danych

T. RZEŹNICZAK

Wiele rozwiązań z zakresu automatyzacji konstruowania prezentacji graficznych koncentruje się na konstruowaniu prezentacji graficznych dla wybranego typu danych, najczęściej bez uwzględniania konkretnej instancji danych w procesie przygotowania metody prezentacji. Wynika to z faktu, że podejścia te starają się być rozwiązaniami uniwersalnymi. W pracy zaproponowana jest metoda, która zwraca uwagę na cechy danych, które faktycznie będą przedstawiane. Znając charakterystykę danych, do zaprezentowania możliwe jest znacznie lepsze dostosowanie podejścia i poprawa wydajności zbudowanej wizualizacji. Wydajność wizualizacji w tym przypadku jest rozumiana jako zdolność obserwatora do odnajdywania i rozpoznawania prezentowanych obiektów w szybki i łatwy sposób. Budowa metody opiera się głównie na badaniach w obszarze wizualizacji danych i psychologii percepcji – ze szczególnym naciskiem na *figural goodness.*

Słowa kluczowe: wizualizacja danych, języki graficzne, structural information theory, figural goodness.